Straightforward IP Integration with IP-XACT RTL-TLM Switching

Marcin Zyś – EVATRONIX - Marcin.Zys@evatronix.pl
Emmanuel Vaumorin – MAGILLEM – vaumorin@magillem.com
Ireneusz Sobański – EVATRONIX - Ireneusz.Sobanski@evatronix.pl

Abstract

This paper gives the results of experimentations done for the packaging of a USB OTG controller respecting the IP-XACT schema provided by The SPIRIT Consortium. It presents the advantages and the technical facts for an IP provider to deliver standardized files package, bringing great advantages for documentation, integration and verification purposes. In particular it is shown how to handle interfaces, registers and multi abstraction level descriptions of the component. This paper also presents the exploitation of the IP_XACT files for the IP testbench and the simulation. The presented work has been realized in the frame of the SPRINT project, funded by the EC.

1. Introduction

1.1. IP-XACT from the SPIRIT consortium

IP-XACT is an XML based open standard defined by the SPIRIT consortium. This non-profit organization provides a unified set of high quality IP-XACT™ specifications for documenting IP using meta-data. This meta-data will be used for configuring, integrating, and verifying IP in advanced SoC design tool sets and interfacing tools using APIs: the LGI[1] and TGI[2] APIs can be used to access design meta-data descriptions of complete system designs. The specification for the schema is tailored to the requirements of the industry, and focused on enabling technologies for the efficient design of electronic systems from concept to production.

1.2. MAGILLEM environment

The Eclipse™ based MAGILLEM Core is at the heart of a fully featured flow optimization suite comprised of an IP-XACT packager, a platform assembly tool, a complete development environment, a flow control tool and a register view kit. The Magillem suite is a one-of-a-kind innovative software tree, enabling flawless and homogeneous design flow integration for various targets such as ASICs, FPGAs, electronic boards, AMS domain and complex systems. Adaptation kits have been developed for each target (Figure 1).

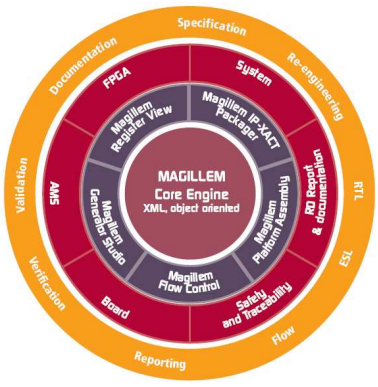

Figure 1 : The MAGILLEM structure dedicated to flow control

[1] LGI: Loose Generator Interface allows SPIRIT database access by file exchanges

[2] TGI : Tight Generator Interface allows SPIRIT database access through a software API

In order to optimize the exploitation of IP-XACT, MAGILLEM 4.1 offers an innovative tooling for IPs import and packaging, design assembly and flow control. It also provides an implementation of the IP-XACT APIs (TGI and LGI), which is required to support point tools encapsulation within a controlled design flow, and complex configurable IPs (most re-usable IPs are configurable). MAGILLEM 4.1 is an environment that enables interoperability and federates existing point tools, languages, methods and models into a seamless, automated flow.

1.3. Evatronix USB On-The-Go IP Core

The Evatronix USB OTG Controller is a fully USB 2.0 specification compliant IP Core that supports On-The-Go supplement. 480 Mbps data transfer is reached in both Device and Host modes, while overall system performance is improved by hardware aid for Session Request and Host Negotiation Protocols. For seamless implementation in majority of SoCs various communication interfaces are available, namely UTMI+, ULPI, PVCI and AMBA™ AHB. With a high level of the controller's configurability users can apply only those features that are essential for the application, thus saving much of the silicon space. Both number, size and buffering of endpoints together with dynamic on-chip memory allocation may help reaching the size target. Store-and-forward or latency buffers architectures and a protocol-aware DMA controller significantly enhance USB data transfer management. The built-in Host Transaction Scheduler enables the CPU to concentrate on writing data and arming buffers. An efficient power management mechanism uses VBUS only during session.

1.4. SPRINT project

The SPRINT project is funded by the European Commission's 6th Framework Program under the IST (Information Science Technology) priority. The project started at the 1st of February 2006 and ends in January 2009. The consortium of the SPRINT project consists of industrial and research partners. Three major European companies are part of the consortium: NXP semiconductor (Netherlands), ST microelectronics (France, Belgium), Infineon (Germany). The global objectives of the SPRINT project is enabling Europe to be the leader in design productivity and quality in Systems-on-Chip (SoC) design, by mastering the SoC design complexity with effective standards and design technology for reuse and integration of IP. The approach is to develop a standards-based open design platform that supports the development of interoperable and reusable IPs and their efficient integration into high quality SoCs. The goal is to stimulate a SoC design ecosystem to the benefit of SoC integrators and (emerging) IP providers and EDA companies and get early availability of advanced SoCs for system integrators to excel in their markets.

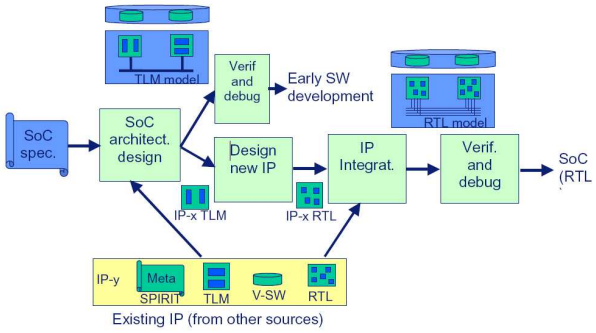

Figure 2: The SoC design flow proposed in SPRINT

The SPRINT project provides a broad and integrated research initiative addressing the complete SoC design flow from initial specification to a verified synthesizable RT-level description of the

SoC. SPRINT advances over a traditional SoC design flow by providing enhanced IP reuse, consistent design across abstraction levels, and enhanced automation of IP integration, verification and debug. As a consequence it supports early verification and debug as well as early software development. This breaks the sequentiality of traditional design flows.

For more information about SPRINT, visit http://www.ecsi-association.org

2. IP-XACT for automating ESL activities

2.1. Overview

IP-XACT from the SPIRIT consortium is nowadays recognized by the electronics community as an apposite choice for managing properly and efficiently the new ESL design flows [1]. Nevertheless, the migration from a legacy design flow to another, taking full benefits of IP-XACT, requires some heavy and complex operations. Figure 3 presents the four steps which have to be completed. They are detailed in the following subsections.

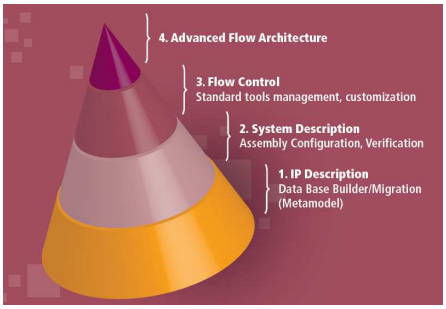

Figure 3: A 4-step methodology to build ESL flows

2.2. IP description

The goal of this first step is to package all the components of a legacy IP library into XML files in accordance with the IP-XACT schema, which describes the syntax and semantic rules for the description of three kinds of elements: the bus/abstraction definitions, the components and the designs (in which components are instantiated). Thus the purpose of the IP packaging is to fill in for each component the XML fields that describe its attributes: physical ports, interfaces, parameters, generics, register map, physical attributes, etc. An important part of the schema is dedicated to referencing the files related to the different views of a component: a view may be for instance a simulable model in a specific language (VHDL, Verilog, SystemC, etc.) or documentation files (e.g. PDF, HTML, Framemaker). This work facilitates future reuse of existing components, because all of their features are easily accessible for its integration and configuration in a bigger system, as it will be explained in the next step. The work that is the topic of this paper takes place in this category: managing the description of the attributes and abstraction levels of a complex and configurable IP using the IP-XACT schema.

2.3. System description

After this step, is it possible to import, configure and integrate components into the system, assemble the design, resolve connections issues, and automate design tasks, thus lightening the verification steps. Some examples of the use of IP XACT at this level are: partial or full automation of design assembly and configuration, detection of communication protocols mismatch, toplevel netlisting, or automatic customization of compilation and simulation of designs. The IP-XACT files describing the hardware platform structure may also be used to perform integration verification based on software, or facilitate the co-design of middleware or software parts of the system.

2.4. Design Automation and Flow control

The third step of the methodology, depicted in Figure 4, aims at linking the design activities around the centric IP-XACT database by means of a dedicated environment which provides access to the IP-XACT information. The tool suite chosen for this study (Magillem environment) provides an IP Packager, a Platform Assembly tool, as well as a Generator Studio to develop and debug additional TGI-based generators. These may be encapsulated within the IP-XACT representation of an IP and may for example simply launch the execution of a script, getting arguments values from the design description in IP-XACT, or be on the contrary a more complex engine, the role of which would be to modify the design itself (e.g. add connections, insert adapters, or configure components).

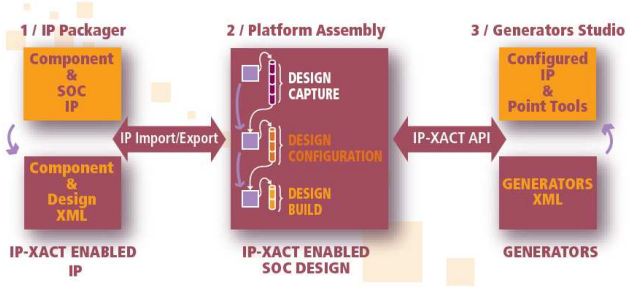

Figure 4: principle schema for an IP-XACT flow

Checkers can also be developed and used to verify design rules at some point, before going further in the design flow. Besides, IP-XACT provides mechanisms to describe the sequences of chained generators and checkers.

2.5. Advanced design flow architecture

This last step in the methodology has a high potential because it exploits all features described previously and allows the actual implementation of advanced ESL activities, such as architecture exploration or software application automated mapping on a hardware platform. These examples show the complexity that has to be managed by the three first steps: all components must be packaged and their configurability must be taken into account; the design assembly automation should be maximized, while any architecture choice should be handled. At last, the generator chains, as defined previously, can be configured and controlled by supervisor engines: for instance a validation sequence will configure and execute several times the generators dedicated to testbench configuration, compilation and simulation.

3. Packaging of the USB core in IP-XACT

3.1. Benefits of the IP-XACT packaging

IP-XACT enables a standard way of IP component packaging and a description of designs at various levels of abstraction. For an IP developer, IP-XACT meta-data is used as a reference and a common base for different design and verification activities. It provides a standard form of component specification and most important enables automation of IP processing. For an IP integrator, on the other hand, IP-XACT standard enables seamless integration of the unknown IPs. The XML file describing the IP contains all the information required for automatic assembly of a SoC. Therefore manual work can be reduced to minimum and the error prone IP

integration task can be significantly simplified. The list of IP-XACT applications is growing, starting from IP verification through refinement and SoC assembly to synthesis.

3.2. Packaging details

The USB-OTG IP is an excellent example of a complex, configurable component packaging according to IP-XACT 1.4 standard. The USB-OTG packaging utilizes various interesting features of IP-XACT 1.4 and shows benefits of IP-XACT usage. Let's depict assets of the most valuable features of IP-XACT 1.4.

Thanks to spirit:busDefinitions and spirit:abstractionDefinition, it is possible to create standard definition of bus interfaces at different abstraction levels. Such definition eliminates ambiguity of different non-standard descriptions, therefore it facilitates interface connection. For the USB-OTG packaging, pre-defined description of AHB and UTMI+[1] bus definitions have been used. For the remaining ports, bus definitions have been created by grouping USB-OTG ports into functional blocks.

Spirit:busInterface specifies the way of mapping the component's ports to the bus. Thanks to that, the IP user does not have to investigate into the functionality of particular component's ports. The bus interface clearly specifies which port corresponds to which bus signal. Taking advantage of the standard and Evatronix-specified bus definitions, the USB-OTG IP-XACT description specifies connection of the IP to these buses. In case of AHB and UTMI+, it can assure automatic connection of the IP in the integrator's system. For the remaining bus interfaces, Evatronix facilitates the connection by providing custom bus definitions. To fully utilize the advantages of bus definitions and interfaces, it is essential for IP-XACT bus definition library to be extended by definitions of all commonly used interfaces. Cooperation between different bus-defining bodies and SPIRIT consortium is essential to prevent users from writing non standard bus definitions.

Spirit:memoryMap provides a consistent way of describing the register set with all the details necessary for software development and basic verification of the component. For each USB-OTG register, its name, address, width, access type, reset value, fields and a short description of usage is provided.

Concept of the spirit:componentGenerators allows to bind the IP-XACT description with external point tools to enable automatic IP configuration, simulation or other IP processing. The RTL view of the USB-OTG is attached with the TGI generator used for IP configuration while making instance of the component.

Spirit:views allows to specify different representation of a component. Exemplary model views are specification documentation, HDL source files and hierarchical design. The concept of a model view is used for automatic RTL/TLM view switching of the USB-OTG.

Spirit:modelParameters and spirit:parameters allow to describe configurability of the component. They can be defined by using such attributes like data format, acceptable values (e.g. a range of values or a list of choices), type of resolution (e.g. a fixed value, a value specified by the user or dependent on another parameter) and a string used to prompt the user. The USB-OTG IP takes advantage of the spirit:parameters and can be fully configured by IP-XACT generators.

Spirit:fileSet is used to reference all files related to the IP: model files in any language, documentation files, or any other file dedicated to a tool and which may be used for the IP. This information is very important because it gives the possibility to actually control the flow by exploiting the files of the IP library.

Spirit:vendorExtension is a feature available in IP-XACT allowing to extend the initial schema for any specific additional attribute, being still compatible with the standard. In the case of the USB-OTG, an Evatronix extension has been used inside the spirit:register to specify that register presence depends on the selected configuration (e.g. the number of USB endpoints). This extension is used only by the USB-OTG generator while generating IP-XACT description of the selected USB-OTG configuration.

3.3. RTL/TLM switching

There is no doubt that TLM emerges as a real standard in ESL modeling. During SoC assembly a common approach is to use transactional models, which are further replaced by corresponding RTL models. To facilitate that process there is a demand for an effortless switching mechanism. This can be achieved by taking advantage of the IP-XACT views, design configurations and abstractors.

In the discussed use case, two views of the USB-OTG transactional model have been defined. One corresponds to the TLM source files (systemCView) and the other is a reference to a design (rtlView), which actually is a hierarchical design containing RTL IP wrapped with interface abstractors.

Both views are identical at the external interfaces. Using spirit:designConfiguration for the design where the USB-OTG is instantiated, it's easy to select the desired view of the USB-OTG. Depending on the selected configuration, either RTL or TLM component is compiled and simulated. This is done by using the same top level environment for both RTL and TLM.

Insertion of abstractors is facilitated by a concept of spirit:abstractionDefinition that defines bus at various abstraction levels. In the design configuration it is possible to insert abstractors at interfaces to adopt bus interface to various abstraction levels.

3.4. Verification Software

An essential task during SoC assembly is the verification of the IP integration. It can be particularly difficult for the third-party or 'unknown' IPs. To verify IP integration it is necessary to at least check the accessibility of the IP registers in the SoC system. It can be done easily by the C integration tests generation from the IP-XACT description of the IP register set. Important requirements for such integration tests are portability (independency of the used platform) and the possibility to use the tests with the transactional model.

To be able to check connection of the interfaces other than system interface, which are not directly accessible by the software, it is often needed additional verification component that can be connected to the integrated IP. For the USB-OTG IP there is UTMI+ VIP[2] that can be connected to the IP UTMI+ interface. The VIP is constructed that way that it is controlled by a set of registers accessible through the system bus. Thanks to the UTMI+ VIP it is possible to perform much more complex tests than just register access check.

To simplify handling of the VIP, a set of predefined functions can be used. These functions operating like simple software stack allows for convenient transmission or reception of USB packets.

4. Experimentations

4.1. Packaging of the USB-OTG IP

The XML description itself can be done in any text editor. However, to both avoid errors and minimize the effort of IP packaging, XML editor aware of IP-XACT schema is highly required, the most

[1] USB Transceiver Macrocell Interface

[2] Verification IP

convenient way being to use the IP-XACT tool dedicated for automation of IP packaging and SoC assembly. In the case of Evatronix USB-OTG IP, Magillem tool supporting the recent IP-XACT 1.4 standard has been used for IP packaging. The Magillem environment was also used for the test bench assembly, the view switching experiments and the verification flow control.

4.2. Instantiation of the USB-OTG component

The XML file provided with USB-OTG RTL IP (i.e. usbhs_otg_rtl_template.xml) is a template of IP-XACT description of the USB-OTG component. Template means here that the description is created for the default value of the component parameters and the register map is described as if the maximum number of endpoint were used. To generate IP-XACT description corresponding to the selected configuration of the IP, an IP-XACT generator is automatically invoked while making instance of the IP selected. The generator is a TGI generator implemented as a java script that retrieves value of the component parameters selected in the IP-XACT tool, and calls the actual configurator implemented as a Perl script. Its goal is to configure the USB-OTG component according to the selected parameters:

- Generation of new 'include file' (i.e. usbhs_otg_mpd_defines.v) with proper value of parameters
- Generation of usbhs_otg_rtl.xml IP-XACT description from the usbhs_otg_rtl_template.xml template description with updated 'memoryMap' and parameter values (e.g. size of AHB address and data buses)
- Referencing the created USB-OTG component, with fixed configuration, to the IP library and creating its instance in the current design

4.3. Simulation environment

TLM hierarchical design was built as a test environment used for RTL/TLM switching and integration verification. The design is very generic and its components are commonly met in SoCs, i.e. CPU model (CPU_BFM), external DMA buffer (MEMORY) and system bus (AHB_BUS). In such environment the USB-OTG component is integrated connecting its AHB master and slave interfaces to the system bus, and interrupt interface to CPU model. Additionally, dedicated verification IP - UTMI_BFM is connected to the USB-OTG UTMI+ interface and to the system bus.

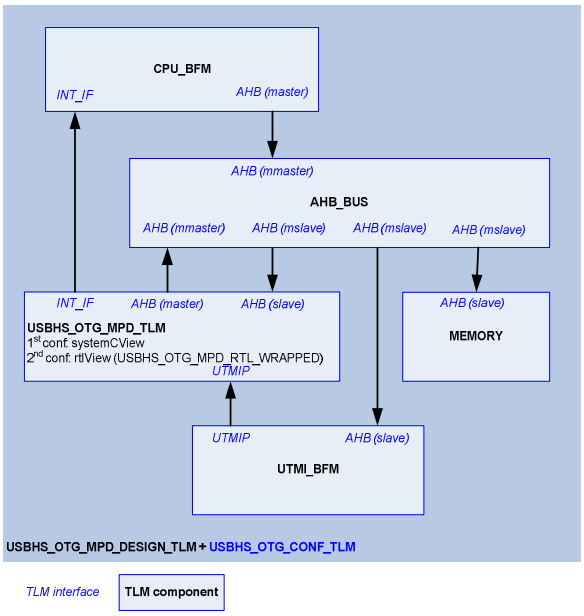

Figure 5: Design using SystemC or RTL view of USB-OTG

For the TLM design there are two configurations prepared. One using USB-OTG TLM and the other using USB-OTG RTL model (see 3.3). For each configuration a generator chain is attached and is used to compile the source files and to run simulation. For the simulation, integration tests are used.

The verification software is implemented as implementation of one of the CPU_BFM processes, which is done in a separate file - cpu_bfm_main_th.cpp.

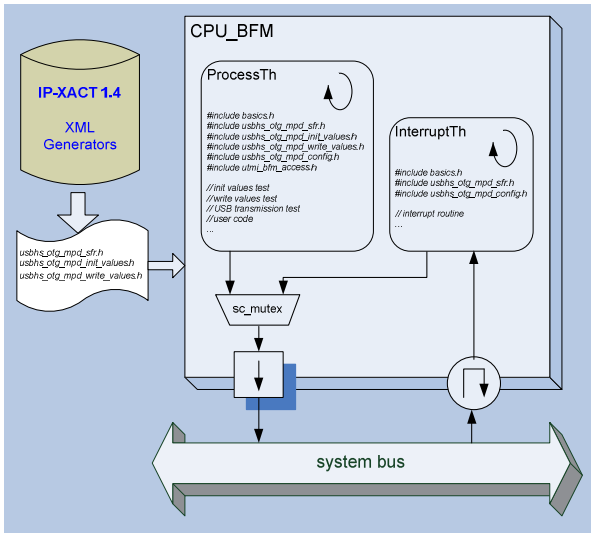

Figure 6: Implementation of verification software in the USB-OTG test environment

IP-XACT description (in particular the component memory map) is used to generate following files:

- usbhs_otg_mpd_sfr.h – definition of the component registers (names) and information related with them (address, reset value, access type)
- usbhs_otg_mpd_init_values.h – test dedicated to check the value of the component registers after reset
- usbhs_otg_mpd_write_values.h – test dedicated to check the value of the component registers after write access

Additionally, to enable more complex tests, high-level functions are defined:

- usbhs_otg_mpd_config.h – functions facilitating configuration of the USB-OTG
- utmi_bfm_access.h/utmi_bfm_access.cpp – functions facilitating usage of the UTMI_BFM

The verification software defined in the cpu_bfm_main_th.cpp takes advantage of all definitions, functions and generated tests. And it is a file that can be further extended by the IP user.

To allow usage of the verification software in various applications (TLM environment, ISS[1], in real hardware as software compiled on a selected processor), functions operating at the lowest level are defined in a separate file - basics.h. Inside that file there are defined macros for basic operations used in the verification software. The selected definition of the macros depends on application.

For instance:

```
/* macro definition for write IO in real system*/
#define WRITE32(addr,data)  *((volatile U32*)(addr)) = (data)

/* macro definition for write IO for TLM_VERIF */
```

[1] Instruction Set Simulator

*extern cpu_bfm * p_cpu_bfm;*

#define WRITE32(addr,data) p_cpu_bfm->WriteLong(addr,data)

An implementation of the interrupt service routines is done in a similar way and is placed a separate SC_THREAD process. Both processes, the main one and the interrupt routine, access the same system bus port of the CPU_BFM. However, access to the port is controlled using sc_mutex.

4.4. Assembly of the RTL/TLM design

A RTL/TLM design has been created in order to wrap USB-OTG RTL IP into transactional interfaces. Simple insertion of abstractors in the design configuration of the test environment was not enough because the current version of IP-XACT design configuration does not allow an interface that exists only at one abstraction level of the component. To work-around this, a separate design is created with two components – USB-OTG RTL IP and the 'dummy' component. Interfaces existing at transaction level are connected to the design interfaces, remaining are connected to the 'dummy' block. And in the design configurations abstractors are put between RTL and TLM interfaces.

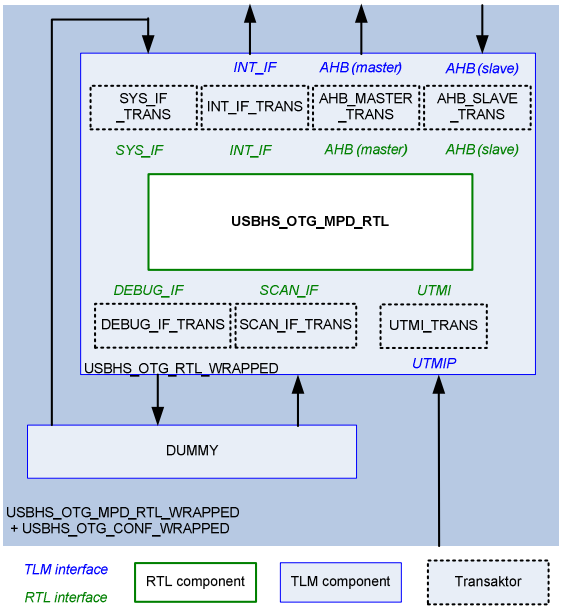

Figure 7: Wrapped RTL IP used as a RTL view of USB-OTG

Thus the created design, which at the external interfaces is equivalent to the USB-OTG TLM IP, is used as an RTL view of the USB-OTG component in the test environment.

4.5. IP package deliverables

A proposed IP package deliverable of the USB-OTG that benefits from IP-XACT usage would consist of:

- RTL IP packaged with IP-XACT 1.4
- TLM IP packaged with IP-XACT 1.4
- Exemplary test environment including UTMI+ Verification IP, all packaged with IP-XACT 1.4
- interface abstractors packaged with IP-XACT 1.4
- RTL IP wrapped with interface abstractors and packaged as RTL view of the USB-OTG IP
- Verification software:
  - definitions and tests generated from IP-XACT description
  - convenience functions for USB-OTG IP and UTMI VIP
  - exemplary verification tests

Such deliverables can reduce to minimum the IP integration effort, enable seamless TLM/RTL view switching and significantly reduce time and cost of verification process.

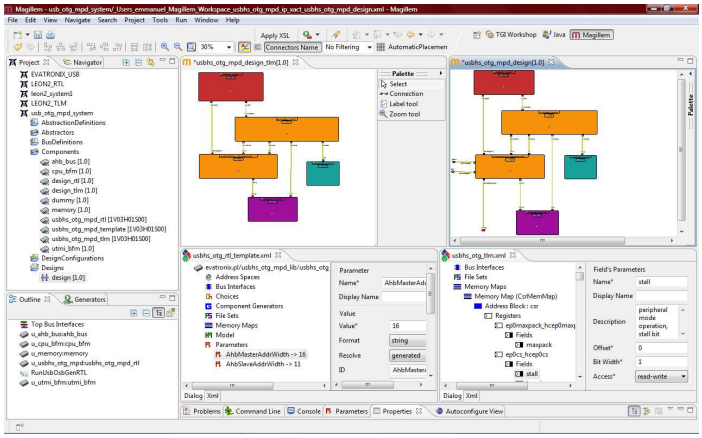

Figure 8: IP-XACT based design in Magillem

5. Conclusion

In this paper we have presented the overall principles allowed by the IP-XACT standard, specified by the SPIRIT consortium, which groups almost hundred major actors of the electronic design industry. By presenting the results of experimentations done around the packaging of a USB OTG core at multi level of abstraction, it has been shown that IP-XACT is a good candidate for IP package deliverable and exploitation. The effort spent on this work represents a few weeks in an experimental frame, which can be seriously reduced to a few days for industrial purpose. The ROI of such effort is important if one considers the time saved for IP integration, system assembly/configuration and validation/verification.

Glossary

EDA: Electronic Design Automation is the category of tools for designing and producing electronic systems ranging from printed circuit boards (PCBs) to integrated circuits. This is sometimes referred to as ECAD (electronic computer-aided design) or just CAD. (Printed circuit boards and wire wrap both contain specialized discussions of the EDA used for those.) The term EDA is also used as an umbrella term for computer-aided engineering, computer-aided design and computer-aided manufacturing of electronics in the discipline of electrical engineering.

ESL: Electronic System Level is an emerging electronic design methodology that focuses on the higher abstraction level concerns first and foremost. It is defined in the book "ESL Design and Verification: A Prescription for Electronic System Level Methodology" by Brian Bailey, Grant Martin and Andrew Piziali and published by Morgan Kaufmann/Elsevier 2007 as: "the utilization of appropriate abstractions in order to increase comprehension about a system, and to enhance the probability of a successful implementation of functionality in a cost-effective manner.

IP: Intellectual Properties are hardware or software modules used to build a system on chip.

IP-XACT: the new name of the SPIRIT schema, which will be standardized as IEEE 1685. Its purpose is to provide a well-defined XML Schema for meta-data that documents the characteristics of Intellectual Property (IP) required for the automation of the configuration and integration of IP blocks; and to define an Application Programming Interface (API) to make this meta-data directly accessible to automation tools.

SoC: A System-on-Chip is an electronic device integrated on a single die.

SPIRIT: initially this acronym (Structure for Packaging, Integrating and Re-using IP within Tool-flows) was used to identify the meta-model schema in XML used to describe RTL and TLM IPs and systems' attributes. This schema is now about to be IEEE standardized and is called IP-XACT. The SPIRIT consortium is the owner of this schema.

TLM: Transactional level Modeling is a high-level approach to modeling digital systems where details of communication among modules are separated from the details of the implementation of functional units or of the communication architecture. Communication mechanisms such as busses or FIFOs are modeled as channels, and are presented to modules using SystemC interface classes. Transaction requests take place by calling interface functions of these channel models, which encapsulate low-level details of the information exchange. At the transaction level, the emphasis is more on the functionality of the data transfers and less on their actual implementation that is, on the actual protocol used for data transfer.

XML: this name stands for eXtensible Markup Language. XML is a markup language much like HTML; it is a file format which was designed to describe data, where tags are not predefined and where you must define your own, through a schema that will be used to set the syntaxic and semantic rules for the files.

Partners presentation

Evatronix

Evatronix SA, headquartered in Bielsko-Biala, Poland, founded in 1991 develops electronic virtual components (IP cores) along with complementary software and development environments that support them. The company also provides electronic design services. Its main design office location, Gliwice (Poland), guarantees easy access to the pool of talented graduates from the Silesian University of Technology. Evatronix IP cores are available worldwide through the sales channels of its strategic distribution partner CAST, Inc. (New Jersey, USA). In the EU countries (excluding UK) and in Switzerland Evatronix operates a direct sales channel. Design services are offered directly by Evatronix worldwide.

Evatronix, as a SPIRIT member, plays its active role by introducing the latest IP-XACT standards in the IP industry. The company aims in implementing the IP-XACT packaging in the IP design and verification processes.

More information on http://www.evatronix.pl

Magillem Design Services has been established by a team of seasoned engineers and a group of business angels in the fall of 2006. The company has inherited "Magillem", a robust and innovative technology worth 120 man years. The headquarters are in Paris, France with a subsidiary in the USA and sales office in Asia. Our team has been a major contributor to the IP-XACT specification since 2003 and we are the only vendor providing a tool implementing all the versions including the latest one of the specification. We audit the existing industrial flows and propose a work plan to adapt them to IPXACT. We validate and verify the full compatibility of tools interfaces into a flow testbench. We test the IP deliverables against a benchmark for compliance using our SPIRIT PACK and check IP integration properties onto a test system. Using our own versatile Magillem toolbox, we are well equipped to offer a wide range of services (Analysing customer's database specifics and customizing SPIRIT Packager accordingly, Implementing new features to meet customer's requirements, Designing complex IP specific configuration nutshells, Designing and implementing system configuration dashboard and custom checkers) to streamlining flow process, developing and integrating specific point tools, developing and integrating tools for the global architecture of a start-to-end ESL flow.

More information on http://www.magillem.com

References

Bailey, B., Martin, G. and Piziali, A. 2007. ESL Design Verification. Morgan Kaufmann Publishers, 2007

Open SystemC Initiative (OSCI) homepage. http://www.systemc.org

SPIRIT Consortium homepage. http://www.spiritconsortium.org

SPRINT Deliverable D5.2c